Self-Adaptive Communication for Collaborative Mobile Entities in ERCMS

Ismael Bouassida Rodriguez*
CNRS; LAAS; 7 avenue du colonel Roche, F-31077 Toulouse, France
Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077
E-mail: ibouassi@laas.fr
*Corresponding author

Jerome Lacouture
CNRS; LAAS; 7 avenue du colonel Roche, F-31077 Toulouse, France
Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077
E-mail: jlacoutu@laas.fr

Khalil Drira
CNRS; LAAS; 7 avenue du colonel Roche, F-31077 Toulouse, France
Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077
E-mail: khalil@laas.fr,

Abstract: Adaptation of communication is required for maintaining the connectivity and the quality of communication in group-wide collaborative activities. This becomes challenging to handle when considering mobile entities in a wireless environment, requiring responsiveness and availability of the communication system. We address these challenges in the context of the ROSACE project where mobile ground and flying robots have to collaborate with each other and with remote human and artificial actors to save and rescue in case of disasters such as forest fires. This paper aims to expose a communication component architecture allowing to manage a cooperative adaptation which is aware of the activity and resource context into pervasive environment. This allows to provide the appropriate adaptation of the activity in response to evolutions of the activity requirements and the changes in relation with the communication resource constraints. In this paper, we present a simulation of a ROSACE use case. The results show how ROSACE entities collaborate to maintain the connectivity and to enhance the quality of communications.

Keywords: Self-Adaptation, Communication, Collaboration

1 Introduction

The ROSACE ("Robots and Embedded Self-adaptive communicating Systems") project aims at studying and developing means to design, specify, implement and deploy a set of mobile autonomous communicating and cooperating robots and software entities with well-established properties, particularly in terms of safety, self-healability, ability to achieve a set of missions and self-adaptation in a dynamic environment. A typical case study considers the context of mobile entities cooperating, into a pervasive environment, in critical operation for crisis management (i.e. to put out fires in a dynamic environment), dealing with heterogeneous and varying ubiquitous communication resources. In this context, communication is essential to achieve the mission objectives since it allows the exchange of information among participants activity. The dynamicity of the environment and the communication resources deployment affect communication integrity. The need to adapt communications is one of the main challenges of the ROSACE project. The communication system is designed as an organisation of autonomous entities which are in charge of managing the communication resources available in the mission in order to provide communication services with the best QoS to actors participating in the operating scenario.

Specific goals of the ROSACE communication system

are:

1. to set up a local network to provide permanent connectivity among ROSACE actors;

2. to manage communication resources to guarantee a permanent connectivity among mission participants;

3. to provide best-adapted quality of service (QoS) according to communication goals and available resources (performance and consistency with activity requirements). For instance, communication priorities have to be taken into account depending on actor's role or the kind of exchanged data.

To achieve the previous goals, adaptation will require to manage the different communication layers (transport and middleware) taking into account the priorities of actors (possibly identified by their roles) and the priorities of the exchanged data. Adaptation should also proceed to modify the behaviour of the involved communicating entities, for example for serving as a communication relay, or more generally for serving to maintain the QoS. This can be achieved by activating predefined functions, acquiring new functions, or delegating to external dynamically discovered services. Managing adaptation calls for the need of monitoring crisis management activities of communication system, and monitoring the supported activity in order to handle the evolution of these requirements. It also requires cooperation among monitoring layers by receiving change notifications and by sending alarms when adaptation is not possible. This paper aims to present the challenge and the solutions for building adaptive communication components in the context of ERCMS (Emergency Response and Crisis Management Systems). Adaptive Communication entities are embedded software components deployed on mobile robots and other communicating devices. They are responsible for maintaining connectivity and quality of communication in group-wide collaborative activities, taking into account mission objectives.

The paper is organized as follow. Section 2 describes the ROSACE project and the case study related to communication adaptation. Section 3 gives an overview of existing communication adaptation techniques and works around autonomic computing. Section 4, we detail the architecture of a node communication manager component providing communication management and adaptation. Then, in section 5, we present how connectivity and QoS preservation is managed thanks to the node communication managers. After than, in section 6, we show simulation results that highlight benefits of our approach. Finally, we conclude.

2 Case study from the ROSACE project

Within the context of the ROSACE project, we define a scenario to enable communication adaptation. The scenario involves various types of mobile actors that carry different types of communication devices such as ground and aerial communicating robots and human actors carrying mobile communicating devices in a wireless communication context. We distinguish human actors that may be professional actors with professional and specific communication devices or occasional actors that carry a mobile device (e.g. PDAs, Phones). We distinguish also, robot actors like planes, helicopters (Autonomous Aerial Vehicle − AAV) and ground robots (Autonomous Ground Vehicle − AGV).

The figure 1 gives an idea of the network organization between the different entities of a ROSACE scenario. Different actors (robots, firemen,...) with different roles (coordinator, investigator,...) are communicating using different types of communication (WiFi, Bluetooth,...) to purchase different goals (fight fire, rescue victims). In this context, the communication objectives are to adapt communication to maintain the connectivity and to propose the best QoS solutions. A main constraint is that, the communication system must deal with expected or unexpected evolution of actors needs and also it has to react to the changes due to device/network constraints.

ROSACE-like activities are based on information exchange between mobile participants collaborating to achieve a common mission. We define three generic roles: the supervisor of the mission, coordinators, and field investigators. Each participant is associated with an identifier, a role and the devices he uses. Each participant performs different functions:

1. The supervisor's functions include monitoring and authorizing/managing actions to be done by coordinators and investigators. The supervisor is the entity which supervises the whole mission. The supervisor waits for data from his coordinators who synthesize the current situation of the mission. The supervisor has permanent energy resources and high communication and CPU Capabilities;

2. Coordinators which are attached to the supervisor, have to manage an evolving group of investigators during the mission and to assign tasks to each one of them. The coordinator has also to collect, interpret, summarize and diffuse information from and towards investigators. The coordinator has high software and hardware capabilities.

3. The investigator's functions include exploring the operational field, observing, analyzing, and reporting about the situation. Investigators also act for helping, rescuing and repairing.

To support groupware activities, network-oriented services should be dynamically activated in response to implicit or explicit requests. These services should provide ubiquitous access to peers and they shloud be technology transparent either wired or wireless. They should take into account different time-varying requirements depending on the targeted activity, users' mobility, exchanged

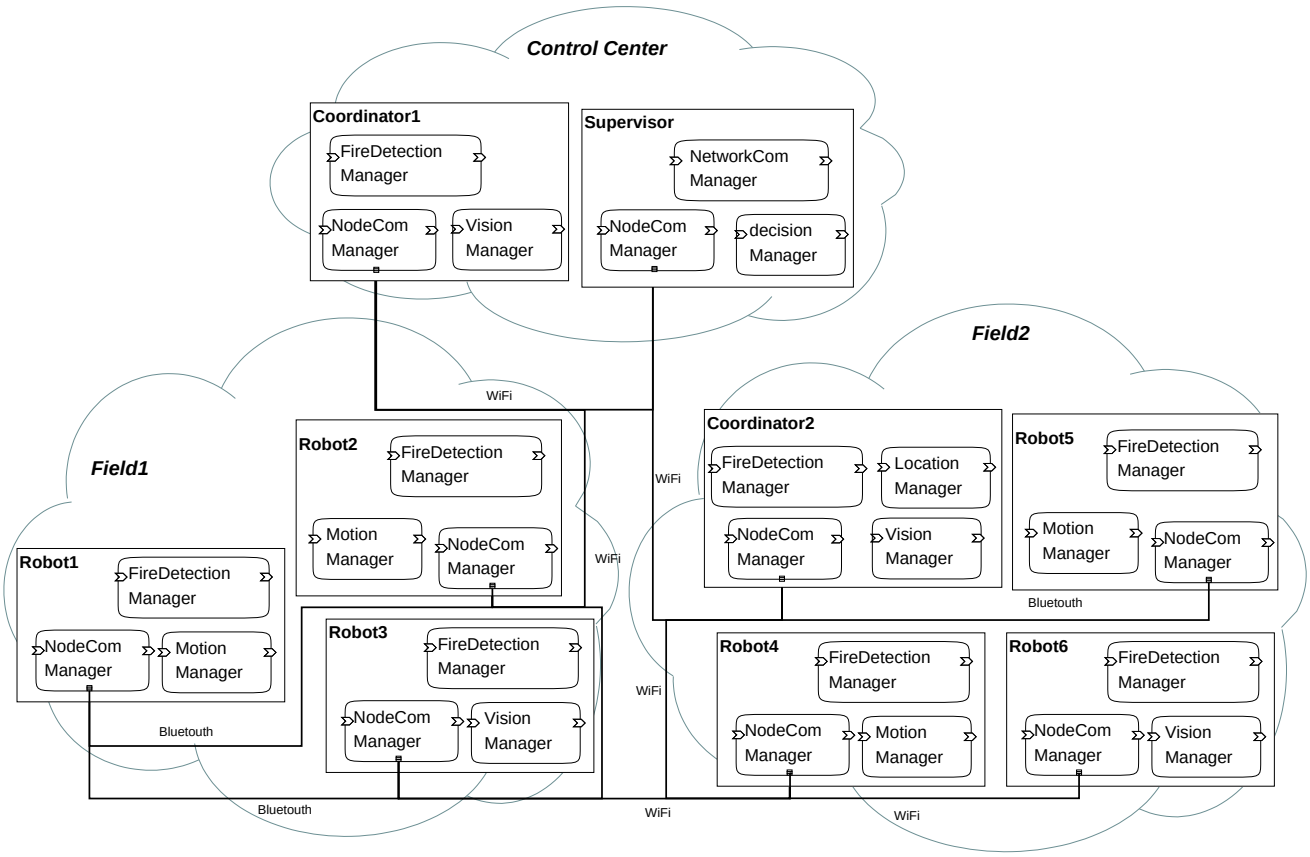

Figure 1: ERCMS architecture

data flows (e.g. audio, video), and different time varying constraints such as variable communication and device resources. Moreover, in ERCMS-like group activities, changes in the cooperation structure between users should also be operated in response to different events such as decisions of the mission's coordinator or informations acquired by the participants. This paper describes an architecture of a communication component to support this kind of adaptive groupware activities.

2.1 Use cases

In ROSACE, we distinguish two main execution steps during a mission: the "Exploration step" (for the localization and the identification of the crisis situation) and the "Action step" (after the event identification). The following scenarios show possible situations where communications can actuate a mission adaptation with different adapted actions. These scenarios include unexpected connectivity discovery (case 1 and 2), expected future loss of connectivity (case 3), loss of connectivity detection (case 4), and a general overview on the management of priorities (case 5). Depending on the case, reactive or pro-active adaptations are considered. Considering the ROSACE action step, firemen, AAV and AGV are deployed and they are achieving their assigned goals.

2.1.1 Use case 1 - Communication device fortuitous discovery

An injured unconscious person equipped with a PDA, a smartphone or such a mobile device (not a device integrating a ROSACE Communication Component (CC)) with activated Bluetooth and/or Wifi interfaces (for instance via his mobile phone) is isolated in a barn/a building near the fire area, or in a trench and so on. Thereof a visual localization appears difficult and this person could escape to visual detection means. A ROSACE actor (i.e. a fireman or a AGV) achieving its goals is getting around the barn or the trench. With an autonomous and transparent way, the ROSACE device of this actor discovers a close unexpected device without disturbing mission performing. The ROSACE global system has to guarantee a localization service of injured persons or humans in danger. As a consequence, following the detection of a new connectivity by a ROSACE actor, the CC of this actor will communicate a notification message to broadcast this event to the control center. Later, the control center will decide to give the responsibility of an observation mission to confirm or not, or to assess, the presence of an injury. The continuation is a "target detection" scenario.

2.1.2 Use Case 2 - ROSACE communication device fortuitous discovery

The same scenario is applicable when the discovered entity is an active communication device carried by a ROSACE actor. This may happen when a new actor joins the group or when a disconnected actor is again in the scope of the communication signal. This situation may occur for example when a fireman has fallen or he is isolated in a building. The group composed of the interconnected ROSACE actors has lost the connectivity with this isolated actor. Then, an entity of another group (e.g. an AGV) is getting around the isolated actor. The ROSACE system has to preserve the safety/integrity of the communication of its actors. As a consequence a similar sequence of actions (see case 1) could be planned. The result is the discovery of a "lost" and "isolated" ROSACE entity and an analysis starts to decide what are possible actions.

2.1.3 Use case 3 - Communication quality preservation/improvement

During the exploration step, a ROSACE group of entities is moving. To maintain the mission, a main goal is to preserve communication within ROSACE entity groups. We assume that a mobile communication entity is equipped with a wifi hotspot and a network is deployed around this hotspot. Another possibility is that a group of AGV (may be in parallel of a wifi infrastructure with hotspots) has built an ad hoc network via wifi interfaces (this network is potentially dedicated to a neighbourhood connectivity supervising). (In either case, each member can locally maintain, periodically, a list of reachable devices, associating to each detected device a level of connectivity quality). In both cases, each entity could detect connectivity quality deterioration between connection interfaces. When a limit threshold value is crossed, a notification message is provided and sent to the control center and also sent to the internal decision making component of the entity. A possible consequence, independent of the message destination localization, is the suggestion to move the entity to preserve communication before a loss of connectivity.

2.1.4 Use Case 4 - Connectivity loss

Alike to the situation of the case 3, a CC detects a connectivity loss with another ROSACE entity (e.g. a robot moves away from the zone where other robots are deployed according to mission requirements). Thus, each of its previous neighbours can notify the network manager entity. Verification steps can be performed by the network manager, sending requests to previous neighbours to find communication relays or testing direct communication with the lost entity, and also locally by previous neighbours, testing other communication interfaces: Bluetooth, infrared, broadcast message for discovery. If the need arises, drones can be allocated to localize the isolated entity, starting the discovery from the last known GPS coordinates.

2.1.5 Use Case 5 - Communication priorities

A complementary scenario deals with the management of priorities. When a critical situation, a fire or an injured person is detected by a fireman or a robot, a relevant adaptation for communications is to give priorities to communication messages of this actor. Priorities will be in relation with the type of message, giving priority to cooperation messages, or the role of participants, giving priority to message from the supervisor to coordinators and from coordinator to investigators.

2.2 A communication component architecture

The presented scenarios show different goals we purchase concerning the management of communications in ROSACE-like environment. They illustrate different facets of communication adaptation that we have to take into account. Use cases 1 and 2 show that we must be able to adapt communications, exploiting them, in relation with mission objectives evolution (for instance, with communication devices discovery). Use cases 3 and 4 highlight the interest to develop on each node (each ROSACE entity) a way to manage connectivity and quality of communication. Finally, the use case 5 shows how priorities can be a first solution of connectivity communication management. In previews works Rodriguez et al. (2010), we have presented a semantic modeling approach to support priorities management and, more generally, to support mobile communicating systems. This first contribution, based on the design of ontologies, SWRL rules and adaptation actions, makes the adaptation aware of the evolving requirements of the activity being supported. In this paper, we present a communication component architecture, using this first contributions. We focus on the description of this adaptive architecture to highlight how adaptation is managed on each node of the network.

3 Related work

In this section, we present our reflections about adapting communication acting on the different communication layers (focusing on the transport and the middleware layers) and proposing an architecture supporting an adaptive mission-aware management. We propose a survey explaining what kind of adaptation we implement (what layer, what type of protocols, what type of monitoring underlining our interest for autonomic computing). This survey also illustrates what kind of activity is managed in ROSACE and highlights that mission and communication are strongly linked. We also present multi-agent systems as our future implementation model. All this elements promotes the contribution proposed in this paper.

On the usefulness of adaptation, M. Satyanarayanan said in Satyanarayanan (2001) "adaptability is necessary when there is a significant disparity between supply and demand of a resource." Adaptation is the operation to make

changes to a program or an information system to maintain its functionalities and, if possible, to improve its performance in a certain execution environment. In the area of ubiquitous computing and context-awareness, adaptation is extended by the concept of adaptability that characterizes a system capability to change its behavior to improve its performance or to continue its role in different environments.

3.1 Goals

After developing an information system, many reasons can lead to adapt it. The reasons for this adaptation can be for corrective, evolutional or perfective purposes Ketfi et al. (2002). ROSACE adresses each of these reasons.

3.1.1 Corrective adaptation

In some cases, we can notice that the application does not behave properly or as expected. The corrective adaptation is a solution to identify the application module that causes the problem and to replace it by a new correct module. This new module provides the same functionality as the former.

3.1.2 Evolutional adaptation

When developing an application, some features are not taken into account. With the changing needs of the user, the application must be extended with new features. This extension can be accomplished by adding one or more modules to provide new features or modifying existing modules to enrich their functionality while keeping the same application architecture.

3.1.3 Perfective adaptation

The objective of this kind of adaptation is to improve application performance. For example, we can have a module that receives a lot of requests and fails to meet them. To avoid system performance degradation, we can duplicate this module to share the requests with the existing one.

3.2 Classification

The adaptation solutions suggested in the literature can be classified as follow.

3.2.1 Design vs run-Time

Two different adaptability views may be distinguished: the design time adaptability Dimka Karastoyanova (2004); Fahmy and Holt (2000); Ermel et al. (2001) and the run time adaptability Chang and Karamcheti (2000); Bade et al. (2006).

For the first view, we can find support tools that handles the application development cycle and optimizes the resources for example. For the run time adaptability Friday et al. (2000) presents several adaptation techniques among which use proxy services, change model of interaction and reorganize application structure.

3.2.2 Local vs Distributed

Adaptation may have a local or a distributed scope. Adaptive components can be deployed on a single machine or distributed on several machines. In the first case, the adaptation is local and only local changes are performed. In the second case, it is distributed and synchronization problems between peer adaptive entities have to be managed Bridges et al. (2001).

3.2.3 Behavioral vs architectural

The adaptation solutions suggested in the literature distinguish behavioral and architectural aspects. The adaptation is behavioral (or algorithmic) when the behavior of the adaptive service can be modified, without modifying its structure. Standard protocols such as TCP and specific protocols such as Wu et al. (2001); Özgür B. Akan and Akyildiz (2004) provide behavior-based adaptation mechanisms. Behavioral adaptation is easy to implement but limits the adaptability properties.

The adaptation is architectural when the service composition can be modified IEEE (2000); Garlan and Perry (1995); Ellis et al. (1996) dynamically. In self-adaptive applications components are created and connected, or removed and disconnected during execution. The architectural changes respond to constraints related to the execution context involving, for example, variations of communication networks and processing resources. They may also respond to requirement evolution in the supported activities involving, for example, mobility of users and cooperation structure modification.

3.3 Objects of adaptation

Adaptation approaches target many levels of information systems: User interfaces, Content, Services, Middleware and Transport.

3.3.1 User interface adaptation

User interface adaptation consists in producing Human-Computer means that can be deployed and used on different types of terminals and which meet the user's preferences. Major existing work in user interface adaptation is based on models that describe the different aspects of the interaction between humans and machines. These models are implemented in different XML or UML languages like UMLi Pinheiro da Silva and Paton (2000) and XIML Puerta and Eisenstein (2002). These models are used to produce the adequate user interface code corresponding to the given XML or UML description. In the existing user interface adaptations, we distinguish two techniques: User Interface transformations and User Interface generation. In the first technique, the adaptation process starts from a description language which is very close to the user

interface code that must be generated. This solution was adopted by [29] to produce adapted Web pages to different terminals starting from an XML description. In this kind of adaptation, style sheets (like XSLT) are used to specify replacement rules of XML tags by scripts that can be directly used by the target device. The second approach consists in generating user interfaces code starting from a high level description which is completely independent from the target programming language of the user interface. SEFAGI Chaari and Laforest (2004) is an example of platforms using the generation technique to ensure user interface adaptation.

3.3.2 Content adaptation

Since the appearance of pervasive systems, the adaptation of multimedia content has been the subject of considerable research. Several techniques for adapting the delivered data to the user have been proposed. These techniques are based on textual transformations Nagao et al. (2001), Housel and Lindquist (1996), image transcoding Wee and Apostolopoulos (2003), or processing video and audio. One of the major issues in content adaptation is where the decision-making and transformations are made. In the literature, three general approaches have been proposed according to the location of adaptation processing between the source that hosts the content and destination that requests it:

1. on the content provider side;
2. on the requester side;
3. at an intermediary (proxy) between the data source and the client.

The content provider-side solutions have some drawbacks. Indeed, the changes made on the content induce a calculation load and consequent resource consumption on the server. However, this approach is very suitable for situations with low variability and low adaptation frequency regarding the simplicity of its implementation. But it is not reliable for cases where the adaptation is triggered frequently.

The content requester-side approach is suitable when the transmission characteristics are less critical than the display limits of the user device Nandagopal et al. (1999). However, the usual complexity of adaptation processing hampers the wide adoption of this approach Perkis et al. (2001). Indeed, the client's terminal has generally very limited computing capacity, power and storage.

For the proxy solution, the flexibility of positioning the adaptation mechanisms on the best content distribution point is a major advantage compared to the other approaches (provider and content sides). However, the proxy must be a trusted party by the provider and the requester of the content. In addition, the third party may charge for the service it provides and the resources it employs to perform the adaptation for the receiver. Therefore, accounting mechanisms should be incorporated in the proxy solution in order to keep track of the amount of resources utilized and the usage of the data.

3.3.3 Service adaptation

Service-Oriented Architecture (SOA) paradigm is based on dynamically publishing and discovering services. This kind of architectures provides the possibilities to dynamically compose services for adapting applications to contexts. Service descriptions are published, via the registry, by service providers and dynamically discovered by service requesters. There are various implementation technologies like COM/ DCOM (Component Object Model) Rogerson (1997); Grimes and Grimes (1997) of Microsoft, the EJB (Enterprise Java Beans) Matena and Hapner (2000); Thomas (1998) of Sun Microsystems or CCM (CORBA Component Model) OMG (1999) of OMG (Object Management Group). We can also consider JXTA Sun et al. (2003) the peer to peer framework or .NET Microsoft (2001) of Microsoft.

3.3.4 Middleware adaptation

Other frameworks are proposed to provide adaptability at the middleware level. In Nasser and Hassanein (2004), an adaptive framework for supporting multiple classes of multimedia services with different QoS requirements in wireless cellular networks is proposed. Sun et al. (2003) proposes CME, a middleware architecture for service adaptation based on network awareness. CME is structured as the software platform both to provide network awareness to applications and to manage network resources in an adaptive fashion.

3.3.5 Transport adaptation

At the transport level, Exposito et al. (2003) provide frameworks for designing Transport protocols whose internal structure can be modified according to the application requirements and network constraints. Adaptation actions correspond to the replacement of a processing module or micro-protocol by another following a plug and play approach.

3.4 Model based adaptation approaches

There are many relevant contributions concerning system architecture adaptation. This kind of approaches uses model-based strategies to apply the necessary transformations on the systems architecture to adapt it to environment and requirement changes. These strategies define or reuse models describing the system software architectures. These models are also known as ADLs (Architecture description languages). We distinguish between there general ADL types: formal ADLs like graph grammars Hirsch et al. (1998) and Petri nets Murata (1989), semantic ADLs using ontologies Zhou et al. (2007) and technical ADLs using XML deployment languages in general Dashofy et al.

(2002). The technical ADLs can be proprietary or implementing the formal and the semantic architecture description models. These ADLs are used to guarantee the architecture evolving and correctness during the different predictable and unpredictable changes in the systems environment. The necessary actions to achieve such adaptations are specified in rules according to the application runtime context. Chaari et al. (2007) is an example of these approaches defining a complete model based architecture adaptation at the Service, content and user interface levels. Chassot et al. (2006) presents another model based method using graph grammars to adapt cooperative information systems to situation changes at the communication level.

3.5 Autonomic Computing

Managing autonomic systems requires to consider abstraction levels. A coordination is needed within and between these abstraction levels. Distinguishing these abstraction levels allows designers and developers to master specification and implementation of adaptation rules. The autonomic concept and systems suggested in the literature are defined in various ways. We target in our study the solutions given within and between differents levels.

3.5.1 Basic concepts of autonomic computing

In this section we present works that focus at the need of autonomic computing and the first concepts are introduced. They focus and detail the autonomic control loop.

For IBM in Kephart and Chess (2003)Autonomic computing systems are those systems that automatically manage themselves by carrying out tasks that have been traditionally performed by computer specialists. The self-management tasks are defined. Self-optimization is the ability of the system to optimize the use of resources. Self-healing is the ability of the system to detect faulty behaviour and self-repair. Self-configuration is the capacity of the system to change its structure and behavior. And self-protection is the ability of the system to detect intrusions, policy violation, etc. and recover from them.

In IBM (2006) the internal functional architecture of an Autonomic Element is shown in this paper. This architecture is composed of a number of functional modules that enable the expected autonomic behavior through a set of autonomic operations. The autonomic operations are achieved using a self-adjusting control loop. Inputs to the control loop consist of various status signals from the system or component being controlled, along with policy-driven management rules that orchestrate the behavior of the system or component. Outputs are commands to the system or component to adjust its operation, along with messages to other autonomic elements.

In Dobson et al. (2006), the authors present a survey of the state of research in autonomic communications and present the autonomic control loop for network communication. They address the five interlinked perspectives of the design and analysis of decentralized algorithms; the modelling, handling and use of context, novel and extended programming approaches; issues and approaches for addressing security and trust; and systems evaluation and testing. They match the challenges such as interaction with stranger, Information reflection and collection, lack of centralized goals and control, meaningful adaptation, cooperative behavior in the face of competition, heterogeneous services and semantics, against the cross-cutting issues and show the technical ideas emerging from each issue when addressing each challenge.

3.5.2 Networks level

In this section we present some works that consider the network point of view of autonomic computing. van der Meer et al. (2006) explains Autonomic Network definition in more detail, links it to the foundational principles of architecture for Autonomic Network Management, and provides guidance on how to develop specifications and best practices for the building of Autonomic Communication Systems. They define a required terminology necessary to support the realization of an Autonomic Communications Framework, and also a flexible framework that can be used as the foundation of autonomic network management. Finally, they specify how this framework can be used to build Autonomic Communications Environments. Four research areas are covered that, in combination, define the foundations of autonomic network management: Modelling and Knowledge Engineering for Autonomic Network Management; Automating Network Configuration via Model-Centred Policy Analysis and Deployment; Network Algorithms and Processes; Architecture and Methodology for Autonomic Network Management.

Agoulmine et al. (2006) gives an overview of the different architectures that support the design, implementation and deployment of autonomic systems. Authors give the motivation behind the emergence of autonomic, self-managed systems and the required features of such architectures. Then, they propose different architectures. They discuss the complexity related to the autonomic information modelling and the autonomic behavior. They present the potential of bio-inspired techniques and compare it with autonomic concepts.

Mullany et al. (2004) examines the trends in next-generation wireless access networks that will lead to the increasing significance of the costs associated with the deployment and configuration of such networks. The authors propose, the concept of a self-deploying, self-configuring radio access network to resolve these issues. They propose algorithms from economic theory, ecology/population growth models, or cellular automata. An example, taken from the field of cellular automata for a radio network capable of self-adaptation to achieve universal coverage in a simplified environment was examined.

In Xiao and Boutaba (2005), the authors present a mechanism for QoS-aware service composition and adaptation of end-to-end network service for autonomic communica-

tion. They introduce a service provisioning framework based on the autonomic communication principle, covering a number of essential functions: domain discovery, domain reacheability, composition, cross-domain contracting, intra-domain provisioning, domain-wide monitoring, and adaptation. Through domain graph abstraction, they reduce the domain composition and adaptation problem to the classic -multiconstrained optimal path problem. They develop a set of new algorithms for QoS-aware service composition and adaptation. Their composition algorithm finds a series of consecutive domains spanning end-to-end and select appropriate service class in each domain such that the overall QoS requirements are satisfied. The algorithm also minimizes the overall cost of the path. As the network condition changes over time or as the user roams across domains, the adaptation algorithm ensures the QoS requirements of the communication path is respected as long as it is feasible to do so, while minimizing the cost of such adjustments. Together, these algorithms are designed to support self-configuration, self-optimization, and self-adaptation of network communication services. They address the service provisioning problem at the domain level, the algorithms can function over heterogeneous intra-domain provisioning mechanisms, and more importantly, provide hard end-to-end QoS guarantees over "soft" intra-domain QoS schemes.

3.5.3 Application level with Agents

In this section we focus in some works that consider a Multi-Agent approach for autonomic systems.

Locatelli and Vizzari (2007) describes a Multi-Agent approach to the modelling and design of Collaborative Ubiquitous Environments, that are environments that support collaboration among persons in a context of ubiquitous computing. In particular, the paper shows how research in the topic of Multi-Agent Systems environment has provided both modelling abstractions and concrete computational supports for the analysis, design and engineering of Collaborative Ubiquitous Environments. In particular, the Multilayered Multi-Agent Situated System model was applied to represent and to manage several types of awareness information (both physical and logical contextual information), which is an essential part of a Collaborative Ubiquitous Environment.

This work differs from other existing proposals that employ agents and agent-based infrastructures simply as a middleware for the design and implementation of pervasive computing systems. The authors present also a selection of the available platforms developed in this context and discuss their suitability to support the development of Collaborative Ubiquitous Environments.

Johnson and Iravani (2007) begins by discussing some of the general issues of complex systems and explains why the agent-based approach is attractive. The article investigates the application of multilevel hypernetworks in team robotics as an example of a complex interaction-based system. The authors show how hypernetworks can represent multilevel relational dynamics by the in-depth analysis of a robot soccer simulation game. They have sketched a mathematical formalism for representing the relational structure between agents.

3.5.4 Frameworks and architectural proposals

In Liu and Parashar (2006), the authors present the Accord programming framework that extends existing programming models/frameworks to support the development of autonomic applications in wide area distributed environments. The framework builds on the separation of the composition aspects (e.g., organization, interaction, and coordination) of elements from their computational behaviours that underlies the component- and service-based paradigm, and extends it to enable the computational behaviors of objects/components/services as well as their organizations, interactions, and coordination to be managed at runtime using high level rules. The operation of the proposed framework is illustrated using a forest fire management application.

In Yechiam Yemini and Florissi (2000), the authors present The NESTOR system. The NESTOR system addresses the needs of network management automation, the complexity of management of Large networks and of minimizing the management of small home networks due to limited resources. The NESTOR system combines several techniques from object modelling, constraint systems, active databases, and distributed systems. In the NESTOR system, managers operate on a unified object-relationship model of the network using a rich set of operations that support rollback and/or recovery of operational configuration states. Declarative constraints prevent known configuration inconsistencies, and in conjunction with policy scripts may automatically propagate changes to maintain consistency. Protocol proxies are used to provide much of this functionality with little or no changes in the network clients. A protocol for replication and distribution of the directory assures availability and operational efficiency. Other works try to analyse Autonomic Computing Systems.

Litoiu (2007) investigates performance analysis techniques used by an autonomic manager. It looks at the complexity of the workloads and presents algorithms for computing the bounds of performance metrics for distributed systems under asymptotic and nonasymptotic conditions, with saturated and nonsaturated resources. The techniques used are hybrid, making use of performance evaluation and linear and nonlinear programming models. The authors treat autonomic transactional distributed systems. The system is modelled with a Queuing Network Model.

Mobile networked systems for the support of group-wide activities have to provide dynamic adaptability for runtime and design-time changes. Such changes are induced by evolving requirements of the supported activities and by evolving resource constraints.

Designing and implementing self-adaptive communicating systems to support such emerging activities is complex.

Mastering this complexity may be achieved by adopting model-based design approaches associated with automated management techniques for dynamic architectural and behavioral adaptability management.

The exposed works allow clarifying the different problems related to self-adaptability. The proposed solutions have different features related to the targeted goals (e.g. QoS, security...), the considered layers (e.g. Application layer, Transport layer...), the adaptation actions (e.g. rate control, congestion control...), the adaptation properties (e.g. architectural, behavioral...), and the way to manage the adaptation and its autonomy.

To face the different kinds of requirements and constraints evolving, behavioral and architectural adaptability is required at several levels simultaneously. This typical cross layering problematic requires managing coordination without which it can lead to performances way below the targeted ones.

For group communication-based cooperative activities, abstraction levels have to be identified. Adaptation at the highest levels should be guided by the evolving of the activity requirements. Adaptation at the lowest levels should be driven by the changes due to device/network constraints.

In previous works Rodriguez et al. (2010), we provide semantic tools (ontologies, SWRL rules, ...) to allow to manage adaptation reacting to needs, priorities and environment changes. The next step, presented in this paper, is to specify and detail the architecture of our communication component manager. The next section details this component which is based on the use of semantic tools.

4 Node communication manager architecture

To support and manage adaptation of communication, following directions presented in the section above and taking into account priorities, we specify a communication component (NCM: Node Communication Manager) encapsulated in each ROSACE entity (ubiquitous device or robots).

Node Communication Manager components (NCM) are thought as high level computing components which encapsulate knowledge and skills about communication infrastructure, networking and transmission technologies. They offer their capabilities to the rest of entities as services which are provided through standard interfaces. In order to achieve service efficiency the NCM will have the following capabilities:

- Mission information awareness, NCM could have access to mission information such as participants, roles, priorities, location and other information which may be used for communication service provisioning;

- Contextual information awareness. This includes: a) The internal context of the underlaying node containing where the NCM is located, and b) the environmental context, where are situated the rest of nodes and NCM's participating in service provisioning;

- Seamless Management of communication resources in hosting nodes to achieve adaptive communication needs. This includes: a) managing internal service APIs and related middleware to configure, set up, and monitoring communication services, b) assessing service quality according to mission needs, c) dynamic re-configuration and management of internal resources to satisfy communication service requirements;

- Cooperation with the rest of NCMs to achieve Organizational goals. This includes: a) achieving its own goals sharing common resources with other NCMs, b) managing internal resources to satisfy collaboration requests, c) interacting with other NCMs to exchange control information and data to guaranty end to end service provisioning and quality control.

NCMs are embedded in mobile robots and other telecommunication devices such as PDA's. They are expected to manage efficiently the overall communication resources in the hosting node where they are deployed. As service providers for other internal units they should report errors and exceptions to their internal client components. If possible, they suggest corrective actions to the control center or decision-making units. Moreover, at the implementation level, a NCM can be viewed as a Multi-Agent System where each internal component would be a role played by a software agent.

The NCM architecture we developed is presented in figure 2. This architecture is composed of a set of component :

- The *DecisionModelManager* component is a high level component introduced to monitor a node. It handles the whole entity according to his decision model. It allows to deliver node status containing services status and resources status. It also decides internal adaptations of communication. For instance, observing a decreased connectivity of a Bluetooth connection, it can be automatically decided to activate a WiFi connection to replace the deteriorated one. Decisions and policies monitored by this component are only communication decisions. When a mission expertise is required, a suggestion can be send to acquire the network manager and the mission expertise.

- The *ComServicesManager* component monitors and gives informations about the quality and the nature of available communication services of the node. Video, audio or messaging services are handled by the ComServicesManager. Actions like increasing the flow ratio are possible. Local adaptations are managed by this entity. The ComServicesManager generates also a service status that reflects available QoS in the Node.

- The *ComResourceManager* component monitors and gives informations about available communication resources (i.e. QoS values like the bandwidth or the lost rate). It collects all the connection status from several connection managers.

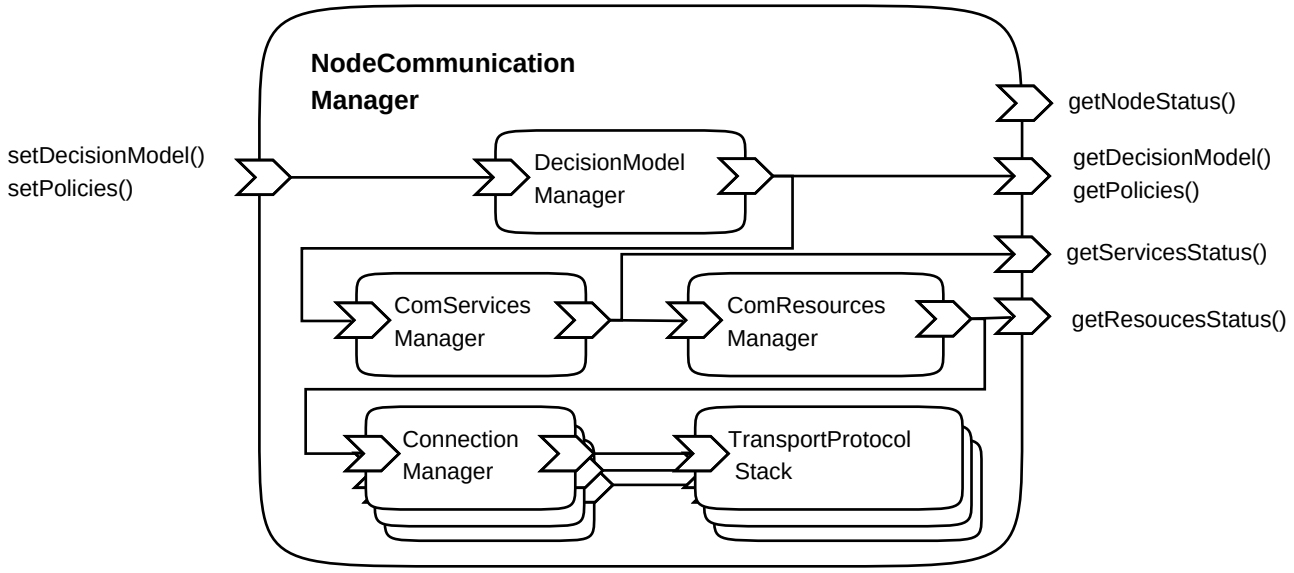

Figure 2: NCM architecture

- The *ConnectionManager* component monitors transport protocol stacks (Wifi, Bluetooth, infrared, GSM) and deliver status to ComResourceManager.
- The *TransportProtocolStack* components manage each protocol stack allowing the different types of communication (Wifi, Bluetooth, infrared or GSM). It generates status from the created links.

5 Connectivity and QoS preservation using NCM

In this section, we show how NCM could produce an efficient adaptation to preserve connectivity or to improve QoS. Our first experimentations refer to use cases presented in section 2. We have considered different situations with communication problems and we have experimented it throw a simulation.

In this paper, we present one of these situations to illustrate NCM benefits. The chosen scenario illustrates different aspect exposed in the presented use cases (connectivity preservation, communication quality improvement, ROSACE communication device fortuitous discovery). Therefore, we consider a group of four ROSACE robots (A ROSACE Robot is a robot equipped with an NCM). The figure 3 gives a summary of the sequence of events of the scenario.

- Step 0. Initially two ground robots (Robot Robot4 and Robot Robot6 (see figure 1) are communicating together. The communication flow between Robot4 and Robot6 is considered as important for their cooperation.
- Step 1. In relation with mission objectives (rescue), Robot6 has to move away. For instance, this step could be initiated if a robot receives a recommandation from the control center to find injured in an area near its position. The ComResourceManager of Robot6 and Robot4 detect a lost rate increase (a policy of adaptation report a problem (connectivity loss) when a threshold of adaptability is reached). A suggestion of adaptation is decided and sent by the decisionModelManager of each robot ("move Coordinator2 to maintain the connectivity").
- Step 2. We consider that Robot4 can't move from his position due to mission constraints. To maintain the communication between Robot4 and Robot6, Coordinator2 has to move near Robot6 (Step 3).
- Step 3. Robot6 is arrived at the required position, Coordinator2 maintains the communication between Robot4 and Robot6. Unfortunately, this communication is not efficient (too important lost rate). The ComServiceManager of Coordinator2 detect this problem. An adaptation policy require to improve this existing link (if possible) when a threshold of comfort is reached. A fortuitous discovery robot (Robot3) can be used as a relay for the communication. Robot3 comes close Coordinator2 and Robot6 and allow to improve the communication link.

6 Simulation Conditions and Results

For the simulation, we consider that Robot4 sends to Robot6 a video flow continuously using MPEG2 protocol. We consider X, a random variable that has a normal distribution with parameters μ and σ which characterizes the

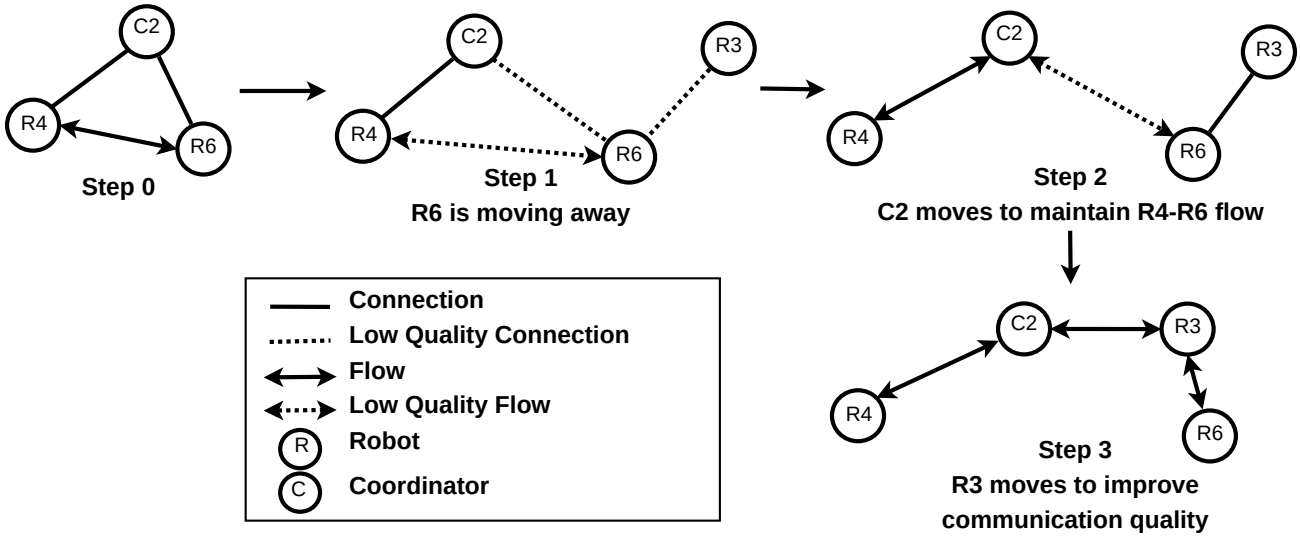

Figure 3: Connectivity and QoS preservation scenario

noise variation on the links. The variation is related to the distance between nodes. The figure 4 gives first results on the observation of one flow (between Robot4 and Robot6). The metric observed to evaluate the connectivity and the quality of the existing flow is the number of lost packets (packet of data that never reaches its destination).

In figure 4, the number of lost packets increases slowly in Step 0. After a minute on the simulation start, Step 1 begins and Robot6 moves to find an injured person in an area near its position. The number of lost packets increases rapidly and reaches the threshold. The process of adaptation is triggered here (see the point A in figure 4) and Coordinator2 moves to avoid connectivity lost. The Step 2 starts, and the number of lost packets decreases in the first minutes of this step. But the number of lost packets still beyond the comfort threshold enough time (1 minute for our simulation) to trigger an adaptation action. Robot3 is used as a relay for communication. Robot3 comes near to Coordinator2 and Robot6. This action is required to enhance the quality of communication. The Step 3 starts and the number of lost packets decreases. Robot6 is arrived at the required position, Coordinator2 and Robot3 maintain the communication between Robot4 and Robot6. It stills a lost of packets but in acceptable level.

Without the adaptation of communication by NCM, many communication concerns still unresolve (connectivity loss (Step 1) or unusable connections (Step 2)). These first results stimulate the development and the deployment of NCM on each node of the network. Using NCMs allows, thanks to a permanent observation of communication activity (in the simulation the observation of the number of lost packets) and the specification of adaptation policy, to adapt communications (NCM allows different types of communication adaptation: switching communication mode, changing quality of communcation services or physical adaptation actions like "moving the robot"). A simulator is currently developed in the scope of the Rosace project to test more complex scenarios with more collaborative and realistic situations.

7 conclusion

In this paper, we have presented an adaptive communication issue in the context of the ROSACE project where mobile actors collaborate, into a pervasive environment, to manage emergency situations such as those occurring during disasters of forest fires. We have identified and discussed the main scenarios where adapting communication is needed for the achievement of the actors mission objectives. We have developed a communication component (NCM) to manage communication adaptation in relation with mission constraints evolution and unpredictable events. This component is distributed in each node of the Rosace network (in each robot, human devices (PDA),...). We have conducted a simulation of a ROSACE use case. This simulation shows the manner that ROSACE entities collaborate to maintain connectivity and enhance quality of communications. In the future, we plan to extend the simulation and involve more entities. The interaction between coordinators and the control center needs to be simulated too. In our work, we consider several connections (Wifi, Bluetooth, etc.). Besides, it will be important to study the cost of using different technology in terms of efficiency.

REFERENCES

Agoulmine, N., Balasubramaniam, S., Botvich, D., Strassner, J., Lehtihet, E., and Donnelly, W. (2006). Challenges for autonomic network management. In *1st IEEE*

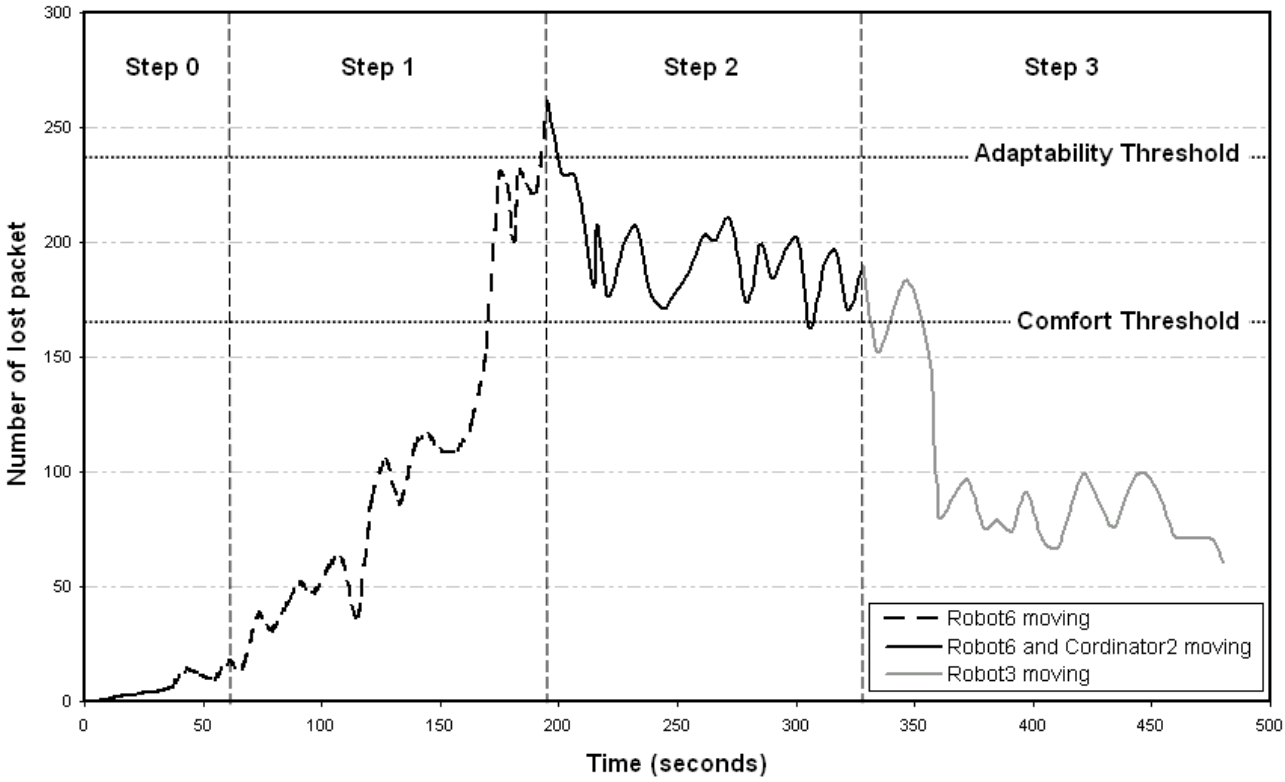

Figure 4: Simulation results using NCM

International Workshop on Modelling Autonomic Communications Environments (MACE).

Bade, K., Luca, E. W. D., Nürnberger, A., and Stober, S. (2006). Carsa - an architecture for the development of context adaptive retrieval systems. In van Rijsbergen, K., Nürnberger, A., Jose, J. M., and Detyniecki, M., editors, *Adaptive Multimedia Retrieval: User, Context, and Feedback*. Springer-Verlag.

Bridges, P., Chen, W.-K., Hiltunen, M., and Schlichting, R. (2001). Supporting coordinated adaption in networked systems. *Hot Topics in Operating Systems, Workshop on*, 0:0162.

Chaari, T. and Laforest, F. (2004). Génération et adaptation automatiques des interfaces utilisateurs pour des environnements multiterminaux projet sefagi (simple environment for adaptable graphical interfaces). *Ingénierie des Systèmes d'Information*, 9(2):11–38.

Chaari, T., Laforest, F., and Celentano, A. (2007). Adaptation in Context-Aware Pervasive Information Systems: The SECAS Project. *Int. Journal on Pervasive Computing and Communications(IJPCC)*, 3(4):400–425.

Chang, F. and Karamcheti, V. (2000). Automatic configuration and run-time adaptation of distributed applications. In *HPDC*, pages 11–20.

Chassot, C., Guennoun, K., Drira, K., Armando, F., Exposito, E., and Lozes, A. (2006). Towards autonomous management of qos through model-driven adaptability in communication-centric systems. *ITSSA*, 2(3):255–264.

Dashofy, E. M., van der Hoek, A., and Taylor, R. N. (2002). An infrastructure for the rapid development of xml-based architecture description languages. In *ICSE '02: Proceedings of the 24th International Conference on Software Engineering*, pages 266–276, New York, NY, USA. ACM.

Dimka Karastoyanova, A. B. (2004). Extending web service flow models to provide for adaptability. In *OOPSLA '04 Workshop on Best Practices and Methodologies in Service-oriented Architectures: Paving the Way to Web-services Success*, Vancouver, Canada.

Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., and Zambonelli, F. (2006). A survey of autonomic communications. *ACM Trans. Auton. Adapt. Syst.*, 1(2):223–259.

Ellis, W., Hilliard, R., Poon, P., Rayford, D., Saunders, T., Sherlund, B., and Wade, R. (1996). Toward a recommended practice for architectural description. In 2^{nd} *IEEE International Conference on Engineering of Complex Computer Systems*, pages 21–25, Montreal, Canada.

Ermel, C., Bardhol, R., and Padberg, J. (2001). Visual design of software architecture and evolution based on graph transformation. In *Uniform Approches to graphical process specification Techniques*, Genove, Italy.

Exposito, E., Snac, P., and Diaz, M. (2003). FPTP: the XQoS aware and fully programmable transport protocol. In *Proc. The 11th IEEE International Conference on Networks (ICON'2003)*, Sydney, Australia.

Fahmy, H. and Holt, R. (2000). Using graph rewriting to specify software architectural transformations. In 15^{th} *IEEE international Conference on Automated Software Engineering, ISBN 0-7695-0710-7*, pages 187–196, Grenoble, France.

Friday, A., Davies, N., Blair, G., and Cheverst, K. (2000). Developing adaptive applications: The most experience. *Integrated Computer-Aided Engineering*, 6(2):143– 157.

Garlan, D. and Perry, D. (1995). Introduction to the special issue on software architecture. *IEEE Transactions On Software Engineering*, 21(4):269–274.

Grimes, R. and Grimes, D. R. (1997). *Professional Dcom Programming*. Wrox Press Ltd., Birmingham, UK.

Hirsch, D., Inverardi, P., and Montanari, U. (1998). Graph grammars and constraint solving for software architecture styles. In *ISAW '98: Proceedings of the third international workshop on Software architecture*, pages 69–72, New York, NY, USA. ACM.

Housel, B. C. and Lindquist, D. B. (1996). Webexpress: a system for optimizing web browsing in a wireless environment. In *MobiCom '96: Proceedings of the 2nd annual international conference on Mobile computing and networking*, pages 108–116, New York, NY, USA. ACM.

IBM (2006). An architectural blueprint for autonomic computing. White paper, IBM Corporation.

IEEE (2000). In *IEEE Std 1471-2000, IEEE Recommended practice for architectural description of software-intensive systems*, pages i–23.

Johnson, J. H. and Iravani, P. (2007). The multilevel hypernetwork dynamics of complex systems of robot soccer agents. *ACM Trans. Auton. Adapt. Syst.*, 2(2):5.

Kephart, J. O. and Chess, D. M. (2003). The vision of autonomic computing. *Computer*, 36(1):41–50.

Ketfi, A., Belkhatir, N., and Cunin, P. Y. (2002). Adaptation dynamique, concepts et experimentations. In *Proceedings of ICSSEA*. In French.

Litoiu, M. (2007). A performance analysis method for autonomic computing systems. *TAAS*, 2(1).

Liu, H. and Parashar, M. (2006). Accord: a programming framework for autonomic applications. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, 36(3):341–352.

Locatelli, M. P. and Vizzari, G. (2007). Awareness in collaborative ubiquitous environments: The multilayered multi-agent situated system approach. *ACM Trans. Auton. Adapt. Syst.*, 2(4):13.

Matena, V. and Hapner, M. (2000). *Applying Enterprise Javabeans: Component-Based Development for the J2ee Platform*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Microsoft (2001). Microsoft .net passport technical overview. Technical report, Microsoft.

Mullany, F. J., Ho, L. T. W., Samuel, L. G., and Claussen, H. (2004). Self-deployment, self-configuration: Critical future paradigms for wireless access networks. In *Proceedings of the 1st IFIP TC6 WG6.6 International Workshop on Autonomic Communication (WAC 2004)*, volume 3457 of *Lecture Notes in Computer Science*, pages 58–68. Springer.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580.

Nagao, K., Shirai, Y., and Squire, K. (2001). Semantic annotation and transcoding: Making web content more accessible. *IEEE MultiMedia*, 08(2):69–81.

Nandagopal, T., Kim, T., Sinha, P., and Bharghavan, V. (1999). Service differentiation through end-to-end rate control in low bandwidth wireless packet networks. In *Proceedings of the 6th IEEE International Workshop on Mobile Multimedia Communications (MOMUC'99)*, San Diego, California, USA.

Nasser, N. and Hassanein, H. (2004). Adaptive bandwidth framework for provisioning connection-level qos for next-generation wireless cellular networks. *Canadian Journal of Electrical and Computer Engineering*, 29(1):101–108.

OMG (1999). Corba components: Joint revised submission. Technical Report orbos/99-02-05, Object Management Group.

Özgür B. Akan and Akyildiz, I. F. (2004). Atl: an adaptive transport layer suite for next-generation wireless internet. *IEEE Journal on Selected Areas in Communications*, 22(5):802–817.

Perkis, A., Abdeljaoued, Y., Christopoulos, C., Ebrahimi, T., and Chicharo, J. F. (2001). Universal multimedia access from wired and wireless systems. *Circuits, Systems, and Signal Processing*, 20(3-4):387–402.

Pinheiro da Silva, P. and Paton, N. W. (2000). UMLi: The unified modeling language for interactive applications. In Evans, A., Kent, S., and Selic, B., editors, *UML 2000 - The Unified Modeling Language. Advancing the Standard. Third International Conference, York, UK, October 2000, Proceedings*, volume 1939, pages 117–132. Springer.

Puerta, A. and Eisenstein, J. (2002). Ximl: a common representation for interaction data. In *IUI '02: Proceedings of the 7th international conference on Intelligent user interfaces*, pages 214–215, New York, NY, USA. ACM.

Rodriguez, I. B., Lacouture, J., and Drira, K. (2010). Semantic driven self-adaptation of communications applied to ercms. In *AINA*, pages 1292–1299.

Rogerson, D. (1997). *Inside COM: Microsoft's Component Object Model*. Microsoft Press.

Satyanarayanan, M. (2001). Pervasive computing: Vision and challenges. *IEEE Personal Communications*, pages 10–17.

Sun, J. Z., Tenhunen, J., and Sauvola, J. (2003). Cme: a middleware architecture for network-aware adaptive applications. In *Proc. 14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 3, pages 839–843, Beijing, China.

Thomas, A. (1998). Enterprise javabeans technology: Server component model for the java platform. Technical report, patrica Seybold Group.

van der Meer, S., Donnelly, W., Strassner, J., Jennings, B., and Foghlú, M. O. (2006). Emerging principles of autonomic network management. In *1st IEEE International Workshop on Modelling Autonomic Communications Environments (MACE)*.

Wee, S. J. and Apostolopoulos, J. G. (2003). Secure scalable streaming and secure transcoding with jpeg-2000. In *ICIP (1)*, pages 205–208.

Wu, D., Hou, Y. T., Zhu, W., Zhang, Y.-Q., and Peha, J. M. (2001). Streaming video over the internet: approaches and directions. *IEEE Trans. Circuits Syst. Video Techn.*, 11(3):282–300.

Xiao, J. and Boutaba, R. (2005). Qos-aware service composition and adaptation in autonomic communication. *IEEE Journal on Selected Areas in Communications*, 23(12):2344–2360.

Yechiam Yemini, A. V. K. and Florissi, D. (2000). Nestor: An architecture for network selfmanagement and organization. *IEEE Journal on selected areas in communications*, 18(5):758 –766.

Zhou, Y., Pan, J., Ma, X., Luo, B., Tao, X., and Lu, J. (2007). Applying ontology in architecture-based selfmanagement applications. In *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, pages 97–103, New York, NY, USA. ACM.